Frictionless Compounding in 24/7 Crypto Markets: A Pilot Study of 2% Per-Trade Targets Using Automated Execution

Author: DJ Billions Research (PI: Almando "DJ Billions" McFadden)

Institution: Colorado Technical University (CTU) — Draft for Academic Review

Date: August 28, 2025

Abstract

This study proposes and evaluates a bot-driven trading methodology to test the speed and feasibility of small, repeated percentage gains in 24/7 crypto markets under zero-fee execution conditions. We examine the **time-to-target** problem: how many 2% profit trades are required to scale initial capital (e.g., USD 1,000) to milestones such as USD 1,000,000 and USD 1,000,000,000 when compounding trade by trade. We contribute an experimental design, data schema, and analysis plan that separate the mathematics of compounding from market frictions (fees, slippage, spread, and signal quality). The pilot focuses on Bitcoin spot testnets to ensure safety and reproducibility, then contrasts "frictionless" scenarios against realistic frictions. We define a **Compounding Speed Index (CSI)**: the number of trades and calendar time required to reach targets at a given net-per-trade return and trade frequency.

Keywords: compounding, algorithmic trading, cryptocurrency, testnet research, execution microstructure, return targets, risk management

---

1. Introduction

Crypto markets operate continuously (24/7), enabling frequent compounding unlike traditional markets with limited hours. Practitioners often target small incremental returns (e.g., 1–2%) and seek to repeat them with automation. While the math of compounding is straightforward, the empirical speed at which such gains can be achieved depends on signal frequency, execution quality, and market microstructure. This paper formalizes the question and proposes an academically sound method to test it without endangering capital or violating compliance constraints.

2. Research Questions

RQ1. Under idealized zero-fee conditions, how many trades at +2% net per trade are required to reach capital targets from a given starting balance?

RQ2. Given an observed signal frequency (trades per day) for a specific rule set, what is the expected calendar time to reach each target?

RQ3. How do realistic frictions (spreads, slippage, non-zero fees) change the number of trades and time to target?

RQ4. What risk controls (position sizing, stop-loss rules) preserve compounding through drawdowns while maintaining throughput (trades per day)?

3. Theoretical Framework

We model balance evolution as:

$$\text{Balance}_n = \text{Balance}_0 \times (1 + r)^n$$

where r is the **net** gain per completed trade and n is the number of trades. Solving for n :

$$n = \frac{\ln(\text{Target}/\text{Start})}{\ln(1 + r)}$$

Examples (Start = 1,000 USD, r = 0.02): - To 1,000,000: $n \approx 349$ trades - To 2,000,000: $n \approx 384$ trades - To 1,000,000,000: $n \approx 698$ trades

Given trades-per-day f , calendar time (days) $\approx n/f$. This is an upper bound assuming entries of adequate quality appear at rate f and achieve r after execution.

4. Related Concepts and Practical Constraints

A. Execution frictions. Bid-ask spread, exchange fees, maker/taker schedules, and slippage move realized r below the target. Zero-fee platforms reduce but rarely eliminate spread and slippage.

B. Signal quality. Increasing trades per day by loosening entry criteria can degrade expectancy (edge), harming r and win rate.

C. Risk and position sizing. Fixed fractional risk (e.g., risk 0.5–1.0% of equity per trade) helps survive losing streaks; sizing interacts with stop distance and volatility.

D. Compliance and testnets. Initial experiments should use exchange testnets or paper trading, with no customer funds and no public marketing of hypothetical performance beyond an academic context and appropriate disclosures.

5. Methods

5.1 Design Overview

Phase 1 (Testnet). Implement a non-custodial, testnet-only bot that executes a transparent rule set (e.g., EMA crossover with RSI filter) on BTC/USDT. Log every signal, entry, exit, and all fills (even partials) with timestamps and prices.

Phase 2 (Friction Experiments). Re-run Phase 1 while injecting controlled frictions: fixed basis-point “fees,” variable slippage models, and actual observed spreads.

Phase 3 (Generalization). Repeat across ETH and a short list of high-liquidity pairs to test robustness.

5.2 Data Schema (CSV or SQL)

- **trades:** trade_id, ts_open, ts_close, side, rule_version, entry_price, exit_price, pnl_quote, pnl_pct, max_adverse_excursion, max_favorable_excursion
- **orders:** order_id, trade_id, ts, type (market/limit), side, price, quantity, status, fee_bps, slippage_bps, spread_bps
- **equity_curve:** ts, equity_quote, drawdown_pct
- **signals:** ts, features (ema_fast, ema_slow, rsi, volatility), decision (enter/skip), reason

5.3 Metrics

Primary: net r per trade, trades per day f , compounding speed n (trades) and n/f (days).

Risk: max drawdown (MDD), Calmar ratio (CAGR/MDD), hit rate, average R multiple.

Execution: realized spread/slippage in bps, fill ratios, latency.

Robustness: performance stability across rule versions and instruments.

5.4 Statistical Plan

- Report means with confidence intervals (bootstrap on trade-level returns).
- Test the difference between frictionless and friction-injected conditions using paired resampling on aligned time windows.
- Present sensitivity curves: n as a function of r and f . Example table: r in [1.8, 2.0]% and f in [1, 10] trades/day.

6. Implementation

We provide two artifacts to support reproducibility: 1. **Paper-trading bot (Python)** that logs trades and equity. 2. **Testnet exchange starter (Node + CCXT)** for Bybit/Binance sandbox accounts, with a minimal web UI to submit market/limit orders and fetch balances.

Operational notes: - Use a VPS and process manager for continuous uptime. - Keep API keys in `.env` server-side only. - Validate symbol availability and min order sizes on testnet.

7. Ethics, Integrity, and Academic Considerations

- This is a methods paper and pilot study. No live investor funds are solicited or accepted as part of the research.
- Any performance shown is hypothetical or testnet-based; it must not be presented as actual, realized investor performance.
- For coursework, acknowledge tool assistance and ensure the final submission reflects your own understanding and authorship, per CTU academic integrity policies.

8. Limitations

- Zero-fee is an idealization; spreads and slippage persist, and liquidity shocks occur.
- Backtest and testnet fills can overstate live execution quality.
- Strategy drift: optimizing for higher trades/day can reduce expectancy.
- Survivorship and look-ahead biases must be guarded against in historical tests.

9. Expected Contributions

- A formal framework to convert micro-level per-trade targets into calendar-time estimates.
- The **Compounding Speed Index (CSI)**: a benchmark for time-to-target under specified r and f .
- An open, reproducible logging schema and testnet stack that students and researchers can extend.

10. Work Plan and Timeline

- **Week 1:** Finalize protocol, configure testnet, dry runs on BTC/USDT.
- **Weeks 2–3:** Collect data continuously, target 1,000–3,000 completed trades under frictionless settings; record f (trades/day).
- **Week 4:** Run friction-injected scenarios (5–20 bps fee/slippage), repeat with same rule set.
- **Week 5:** Analyze, bootstrap CIs, produce CSI curves and tables.
- **Week 6:** Write-up results, internal review, and submit to CTU as a methods paper with appendices and code links.

11. Appendices

A. Core Formulae

- Compounding trades: $\text{Balance}_n = \text{Balance}_0(1 + r)^n$
- Trades needed: $n = \ln(\text{Target}/\text{Start})/\ln(1 + r)$
- Days needed: $\text{Days} = n/f$

B. Example Targets at $r = 2.00\%$ (net)

- Start 1,000 USD → 1,000,000 USD: $n \approx 349$

- Start 1,000 USD → 2,000,000 USD: $n \approx 384$
- Start 1,000 USD → 1,000,000,000 USD: $n \approx 698$

C. Risk Control Checklist

- Fixed fractional risk per trade (0.5–1.0% of equity)
- Hard stop-losses; no averaging down
- Venue whitelist and outage playbook (kill-switch on exchange errors)
- Daily reconciliation and equity curve monitoring
- Parameter change log for rule versions

D. Reproducibility Artifacts

- **Paper trading bot:** `btc_bot.py` with CSV logs (`trades.csv`, `equity.csv`).
- **Testnet server + web UI:** routes for ticker, balances, orders, cancels.
- `.env` templates with no secrets included.

Acknowledgments. Prepared by DJ Billions Research as a draft for academic review at CTU.